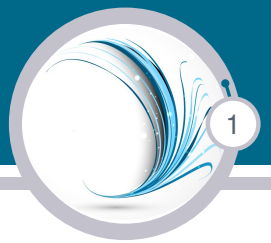
Modeling Rational Agents with a Genetic Algorithm

A spyGEM Computational Model (and some more stuff)

Matthia Sabatelli

August 11, 2016

About the Presentation



- Modeling the potential behavior of *spyGEM* Users:
  - Sender
  - Criminal
- Multi-Agent System, based on:
  - Epistemic Logic
  - Machine Learning
- Using Machine Learning to compute the optimal features of the architecture, *Genetic Algorithm*
- Outcome of the model tells us how to build *spyGEM* in the optimal way

Introduction

- *spyGEM* was born as a new method to **store** and **send** highly reserved information:
  - Advanced Encryption Standard Method
  - Biologically inspired security layers
- Will this be enough to have a successful product at the end?
  - *What happens if some of the Agents don't know how to exactly behave while using spyGEM?*
  - *What happens if the Criminal knows some or all of the information required to make spyGEM fail?*
  - *Will spyGEM still be as good as we thought?*
- **Computational Model** might help us!

Architecture of the Model

- Which features might be important to model while considering a potential *Sender* & *Criminal*? How to represent them in the simulation?
  - 3 BDI components [1] + additional parameters
  - Chromosomes
- Rao et al. [1] state that every rational action is based on:
  - Beliefs
  - Desires
  - Intentions
- Main idea is to identify which of the three BDI components is considered as more important by the Agents!

Architecture of the Model

- Beliefs: are defined as those worlds that the agent believes to be possible since there are no restrictions that can avoid an Agent to believe in something. This feature is formalized as follows: $\forall$ w $\in$ W.
- Desires: are those worlds that correspond to the optimal future scenario an agent wants to achieve: $\exists!$ w $\in$ W.
- Intentions: are considered as sets of intention-accessible worlds. These worlds are ones that the agent has committed to attempt to realize. There is a strong component of realism since the worlds that are part of this feature are only the ones that the Agent thinks can be a possible *Desire*.

Architecture of the Model

- The 3 components are useful but not enough to train a Machine Learning algorithm, every Agent has three more sub-components that influence its actions:
- *Sender:*
  - γ: which corresponds to a particular level of *riskiness* the agent will take into account in his actions.
  - ψ: is a variable that is related to the level of *efficiency* that the agent wants to pursue in its actions
  - α: corresponds to the *time resources* the agent is willing to invest when creating its BDI features

Representation of the Sender

- An Agent is created as a string of binary bits and is represented as a chromosome, each part of it represents the 3 BDI features together + 3 additional parameters

$$Chromosome_1 = \underbrace{010010011010}_{Beliefs}\underbrace{011101110101}_{Intentions}\underbrace{010100101010}_{Desires} \quad (1)$$

$$Beliefs = \underbrace{0100}_{\gamma}\underbrace{1001}_{\psi}\underbrace{1010}_{\alpha} \quad (2)$$

- Every base-pair represents a decimal number which is later on used for computing the fitness value of the Agent and to train the *Genetic Algorithm*.

Genetic Algorithms

- Machine Learning technique inspired by natural evolution
- Different solutions to a potential problem are computed, the ones that are closer to it are kept and give rise to new ones while the "weakest" ones are discarded
- **Only the final optimal solution is known but not the ways it can be reached!**

The Sender Model

$$f(x) = \sum_{i=0}^{n} x[i] = 60 \tag{3}$$

- 20 Simulations
- 20 Pools of 100 Chromosomes: the simulation starts with 20 randomly generated chromosome pools with 100 agents each
- Truncation Method: is the approach that has been used in order to keep the fittest agents allowed to reproduce themselves (70%)
- One point single crossover: is the breeding method that has been used to train the agents

The Sender Model

$$Parent_1 = \underbrace{\mathbf{010}}_{CrossoverBits} 01001101001110111010101010010101 0 \quad (4)$$

$$Parent_2 = \underbrace{\mathbf{011}}_{CrossoverBits} 01000001001110111100101011101000 \quad (5)$$

$$Child_1 = \underbrace{010}_{Parent1Bits} \underbrace{01000001001110111100101011101000}_{Parent2Bits} \quad (6)$$

$$Child_2 = \underbrace{011}_{Parent2Bits} \underbrace{01001101001110111010101010010101 0}_{Parent2Bits} \quad (7)$$

1% mutation rate to avoid local minima: one chromosome is randomly changed from 0 to 1 or vice-versa.

Sender Model Results

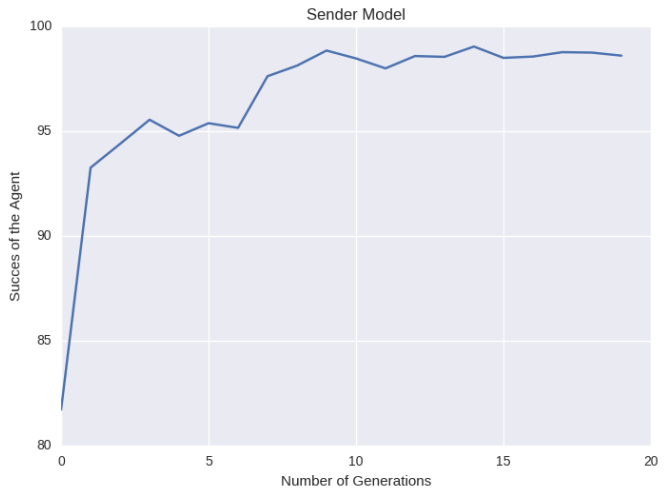


Figure: The graph shows how after 20 generations of breeding the pool of chromosomes representing the *Sender* agents converge to the optimal result. Even though some local minima can be identified.

Figure: Histogram with the analysis of the BDI features. Different colors represent different BDI components: Beliefs are shown in yellow, Desires in blue and Intentions in red.

The Criminal Model

ψ: Amount of knowledge the agent has about the different *Time Trees*
α: Amount of motivation and effort he is willing to put while interfering with the *Sender's* plan

$$f(x) = \sum_{i=0}^{n} x[i] = 65 \tag{8}$$

- 20 Simulations
- 20 Pools of 100 Chromosomes: the simulation starts with 20 randomly generated chromosome pools with 100 agents each
- Truncation Method: is the approach that has been used in order to keep the fittest agents allowed to reproduce themselves (80%)
- Double point crossover: is the breeding method that has been used to train the agents

The Criminal Model

$$Parent_1 = \underbrace{\mathbf{010}}_{CrossoverBits_1} 01001101001110111010101010010 \underbrace{\mathbf{010}}_{CrossoverBits_2} \quad (9)$$

$$Parent_2 = \underbrace{\mathbf{011}}_{CrossoverBits} 01000001001110111100101011101 \underbrace{\mathbf{000}}_{CrossoverBits_2} \quad (10)$$

$$Child_1 = \underbrace{010}_{Parent1Bits} \underbrace{01000001001110111100101011101}_{Parent2Bits} \underbrace{010}_{Parent1Bits} \quad (11)$$

$$Child_2 = \underbrace{011}_{Parent2Bits} \underbrace{01001101001110111010101010010}_{Parent1Bits} \underbrace{000}_{Parent2Bits} \quad (12)$$

Criminal Model Results

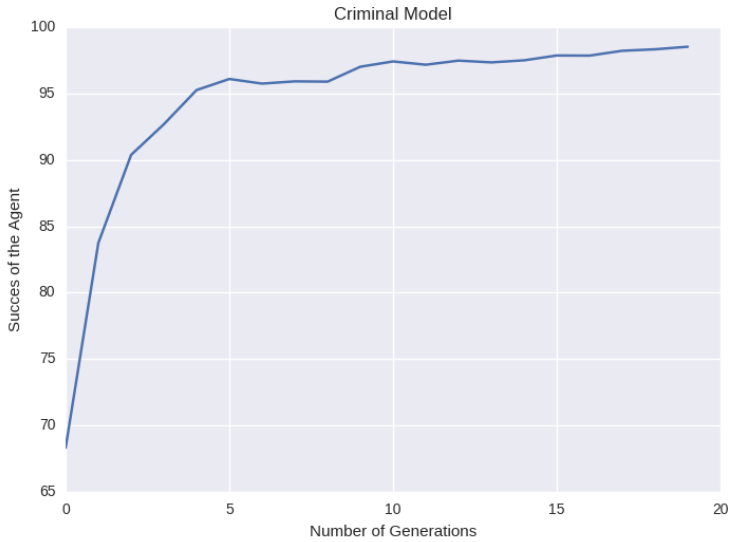


Figure: The performance of the genetic algorithm.

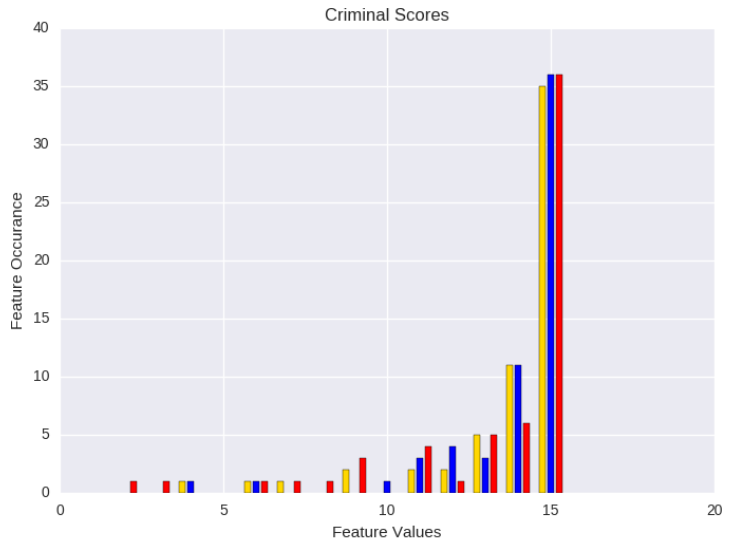
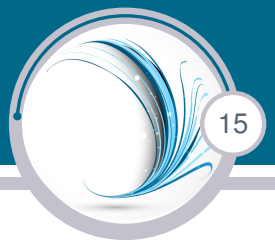


Figure: The analysis of the BDI features

Discussion

- The simulation shows how it is possible to model Artificial Agents with a *Genetic Algorithm* as explained in [2,3,4]
- The model also shows how strong the *spyGEM* system is:
  - A *Sender* has much less knowledge to consider if he wants to be successful, (Belief feature that is discarded) For the *Criminal* this is not true! All the three features are equivalently important. Adding the list of security layers is the correct strategy to make life as hard as possible to the *Criminal* since he has to consider as many *Worlds* as possible.

DNA Model



- **Exploring the impact of Random Mutations**
- Downloaded *Bacillus subtilis* DNA and its exact sequence = **4.215.619** base-pairs
  `https://www.patricbrc.org/portal/portal/patric/Downloads?cType=taxon&cId=224308`
- Our message is $\approx$ **556** base pairs long
- The random mutation rate in the genome is $\approx 5\%$

DNA Model

- In 100 simulations with the standard length of the message, the amount of times a random mutation occurred on the message is **0!**
- It is possible to increase the size of the message exponentially for 5 times (n) so that it reaches a total length of **66720** and still no random mutations damage the message!

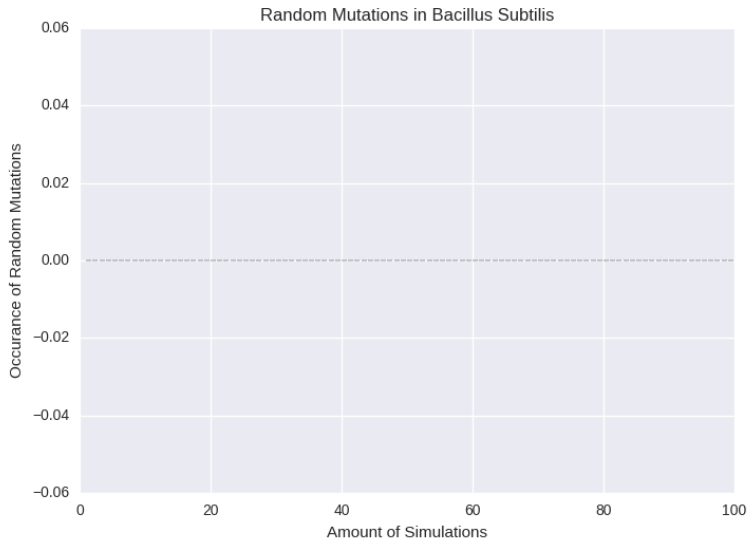


Figure: Random Mutations Occurance

DNA Model

$$f(i) = 556 \prod_{i=1}^{N} i \tag{14}$$

With N=6 the first 2 random mutations occur on a message with a length of **400320** base-pairs.

References A.I. Model

1. Rao, A. S., & George, M. P. (1991). Modeling Rational Agents within a BDI-Architecture.
2. Herrera, F., & Verdegay, J. L. (1996). Genetic algorithms and soft computing. Physica-Verlag.
3. Zhong, W., Liu, J., Xue, M., & Jiao, L. (2004). A multiagent genetic algorithm for global numerical optimization.
4. Goldberg, D. E., & Holland, J. H. (1988). Genetic algorithms and machine learning. Machine learning, 3(2), 95-99.

References DNA Model

1. Drake, J. W., Charlesworth, B., Charlesworth, D., & Crow, J. F. (1998). Rates of spontaneous mutation. Genetics, 148(4), 1667-1686.
2. Shafikhani, S., Siegel, R. A., Ferrari, E., & Schellenberger, V. (1997). Generation of large libraries of random mutants in Bacillus subtilis by PCR-based plasmid multimerization. Biotechniques, 23(2), 304-311.
3. Kunkel, T. A., & Bebenek, K. (2000). DNA Replication Fidelity*. Annual review of biochemistry, 69(1), 497-529.

Last slide :)

- A.I. Model: `http://2016.igem.org/Team:Groningen/AIModel`
- A.I. Software: `https://github.com/paintception/myGEM`
- DNA Model Software: `https://github.com/paintception/spyGEM2016`